Sizing, shaping, and topology optimization of truss-structure using ant algorithms

Guan-Chun Luh **Chun-Yi Lin**

Department of Mechanical Engineering,

Tatung University, Taipei, Taiwan, R.O.C.

Abstract

In this study, a two-stage ant algorithm consisting of the Ant System and the API (after *Pachycondyla* API*calis*) algorithm was employed for finding optimal cross-sectional size, topology, and shape truss-structure to achieve minimum weight objective under stress, deflection, and kinematic stability constraints. The effectiveness of the proposed ant algorithm is evaluated through a two-tier, 12-node, 39-member ground structure.

Keywords: ant algorithm, Ant System, API algorithm, truss-structure optimization

1. Introduction

Optimal design of truss-structures has always been a fast developing area of research in the field of engineering optimization and has made notable progress in the last decade. Optimization of truss-structures can be classified into three main categories: sizing (cross-sectional area), shaping (configuration), and topology optimization. Apparently, the most efficient way to design truss structures optimally is to consider all three schemes simultaneously.

Two kinds of approaches have been used to solve such problem in the literatures. One is a double stage method; first, the topology of the structure is optimized from a given ground structure which contains a large set of candidate trusses, and next, the size as well as the shape of each member are optimized. The other is the singular stage method; all design variables including topology, shape, and size are integrated as one design variable. Genetic algorithms [1], simulated annealing [2] and genetic programming [3] have been employed for optimum design of truss type structures with respect to size, shape and topology design variables simultaneously.

Deb and Gulati [1] employed real-coded genetic algorithms for finding optimal cross-sectional size, topology, and configuration of 2-D and 3-D trusses to achieve minimum weight under stress, deflection, and kinematic stability constraints. They used a representation scheme that naturally allows all three optimizations to be used simultaneously. The proposed algorithm has been able to find trusses better than those reported in the literature in a number of different truss-structure problems. In addition, they showed that there existed many different topologies with almost equal overall weight in truss-structure design problems. In other words, the resulting solution of truss-structure optimization design problem is multi-modal [1].

A two-stage ant algorithm, consisting of the Ant System (AS) [4] and the API (after *Pachycondyla* API*calis*) [5] algorithms, was proposed in this study for finding optimal cross-sectional size, topology, and shape truss-structure to achieve minimum weight objective under stress, deflection, and kinematic stability constraints. The AS is a relatively recent approach to solve combinatorial optimization problems mimicking the behavior of real ant colonies. It has been successfully applied to solve a number of different optimization problems such as the Traveling Salesman problem and the Vehicle Routing Problem. Note that all of these problems involve the ordering of discrete number terms and most of the works in the area of AS has concerned discrete optimization problems. Comparative less work has applied the ant colony metaphor to continuous space optimization. The API algorithm developed inspired by a model of the foraging behavior of a population of primitive ants named *Pachycondyla apicalis* is one of them.

The proposed Ant algorithm (*i.e.* AS-API) is a good candidate for truss structure optimization due to its discrete-continuous combination feature. However, not research has been done so far. Recently, Camp and Bichon [6] applied ACO algorithm optimal design cross-sectional areas (discrete variables) of space trusses. It was illustrated that their proposed ACO algorithm could design truss structure satisfying constraints while minimizing the overall weight.

2. Ant algorithms for truss structure optimization

2.1 Problem definition

Given a set of supports, concentrated loads, and possible node points in a structural domain, the

problem is determining the optimal element connectivity, member sizing and node positions that results in a least weight structure satisfying prescribed design constraints. The formulation of the optimization problem adopted in this study for comparison [1] can be described as follows:

$$\text{Minimize} \quad W(\boldsymbol{A},\xi)=\sum_{j=1}^{n_m}\rho_j \ell_j A_j$$

$$\begin{aligned}\text{Subject to} \quad & G_1 \equiv \text{Truss is acceptable to the user},\\ & G_2 \equiv \text{Truss is kinematically stable},\\ & G_3 \equiv S_j-\sigma_j(\boldsymbol{A},\xi)\ge 0, \qquad j=1,2,...,n_m,\\ & G_4 \equiv \delta_i^{\max}-\delta_i(\boldsymbol{A},\xi)\ge 0, \qquad i=1,2,...,n_n,\\ & G_5 \equiv \boldsymbol{A}_j^{\min}\le \boldsymbol{A}_j\le \boldsymbol{A}_j^{\max}, \qquad j=1,2,...,n_m,\\ & G_6 \equiv \xi_i^{\min}\le \xi_i\le \xi_i^{\max}, \qquad i=1,2,...,n_n.\end{aligned}$$

where the design variable $\boldsymbol{A}$ and ξ denote the cross-sectional areas of n_m members and the coordinates of all n_n existing nodes present in the truss, respectively. The parameters S_j and $\delta_i^{\max}$ indicate the allowable strength of the jth member and the allowable deflection of the ith joint defined by the designer, respectively. The parameters ρ_j and ℓ_j are the material density and length of jth member.

Constraint G_1 checks if the truss structure has all the basic nodes. Constraint G_2 is employed to check if the truss is kinematically stable. Similar to [1], the Grubler's criterion [7] is used to check the degree-of-freedom (DOF) of the truss at first. Then stiffness matrix is used to check if it is positive-definiteness. Constraints G_3 and G_4 are implemented to check the stresses of all n_m members and deflections of all n_n nodes. For a feasible truss, all members must have stresses within the allowable strength of the material and all nodes must not deflect more than the allowable limit due to the applied loads. G_5 and G_6 indicate that the design variables are bounded.

The objective value $f^k(\boldsymbol{A},\xi)$ for the kth solution/truss topology is illustrated as follows [1]:

$$f^k(\boldsymbol{A},\xi)=\begin{cases}10^9 & \text{if } G_1 \text{ is violated}\\ 10^8 & \text{if } G_2 \text{ is violated with DOF constraint}\\ 10^7 & \text{if } G_2 \text{ is violated with the positive - definiteness constraint}\\ W^k(\boldsymbol{A},\xi)+10^5\cdot\sum_{i=1}^{n_m}\left|\langle G_{3i}\rangle\right|+10^5\cdot\sum_{j=1}^{n_n}\left|\langle G_{4j}\rangle\right| & \text{otherwise}\end{cases}$$

where the operator $\langle\bullet\rangle$ is the bracket-operator penalty term.

2.2 AS algorithm

The AS algorithm for truss topology optimization follows the following steps:

(1) *Initialization.*

A population of m ants is placed randomly in the n nodes. The cross-sectional area of the jth member is set to their average values $(\boldsymbol{A}_j^{\min}+\boldsymbol{A}_j^{\max})/2$.

(2) *Node transition rule.*

The transition probability from node i to node j for the kth ant is defined as:

$$p_{ij}^k(t)=\begin{cases}\dfrac{[\tau_{ij}(t)]^\alpha}{\sum_{h\notin \boldsymbol{tabu}}[\tau_{ih}(t)]^\alpha} & \text{if } j\notin \boldsymbol{tabu}\\ 0 & \text{otherwise}\end{cases}$$

where ***tabu*** is used to define the set of nodes have been visited for all ants in last period (t-1).

(3) *Niche strategy.*

Niche strategy is utilized to find the multiple solutions since truss structure optimization is a kind of multi-modal problem. In this study, sharing scheme is employed to calculate the similarity for each solution/topology and described below.

$$f_{share}^k(\boldsymbol{A},\xi)=\frac{f^k(\boldsymbol{A},\xi)}{\sum_{j=1}^m s_j^k}$$

where $f^k(\boldsymbol{A},\xi)$ and $f_{share}^k(\boldsymbol{A},\xi)$ denote the original objective and the shared objective value of the kth ant, respectively; s_j^k indicates the similarity between the kth ant and the jth ant,

$$s_j^k=\frac{\sum_{i=1}^M\left(M_i^k\ \mathbf{XNOR}\ M_i^j\right)}{M}$$

where M denotes the total number the members in the ground structure; M_i^k and M_i^j indicate if the ith member exists for the kth and jth solution/truss topology, respectively.

$$M_i^k;M_i^j\equiv\begin{cases}0 & \text{if the } i\text{th member is absent}\\ 1 & \text{if the } i\text{th member is pressent}\end{cases}$$

The binary **NXOR** operator has the following features: 0 **NXOR** 0 =1 and 1 **NXOR** 1 = 1. $0\le s_j^k\le 1$. $s_j^k=1$ means that the truss topologies constructed by the jth and kth ants are identical while $s_j^k=0$ means those are completely different.

(4) *Pheromone updating rule.*

The trail intensity is updated according to the following formula

$$\tau_{ij}(t+n)=(1-\rho)\tau_{ij}(t)+\Delta\tau_{ij}+e\cdot Q/f^*$$

where e indicates the number of elitist ants, Q is a positive constant value, and f^* is the best objective value of solution/topology found from the beginning of the trail. In an effort to improve performance, "elitist ants" (similar to the elitist strategy used in genetic algorithm) introduced by Dorigo *et al.* [4] is included in the pheromone updating rule.

$$\Delta\tau_{ij}=\sum_{k=1}^m\Delta\tau_{ij}^k$$

$$\Delta\tau_{ij}^{k}=\begin{cases} Q/\ f_{share}^{k}(\boldsymbol{A},\xi) & \text{if the } k\text{th ant uses edge } (i,\ j) \text{ in} \\ & \text{its tour between time } t \text{ and } t+n \\ 0 & \text{otherwise} \end{cases}$$

(5) *Memories of multiple colonies*.

In this study, multiple-colony memories were employed to save the multi-modal topologies found. The number of memories n_m was predefined and the solutions saved in memories will be replaced by the best solution found currently if it satisfies the following conditions:

$$f^{k}(\boldsymbol{A},\xi) > f^{memory}(\boldsymbol{A},\xi) \ \cap \ \ s_{memory}^{k} \geq \delta$$

where $f^{k}(\boldsymbol{A},\xi)$ and $f^{memory}(\boldsymbol{A},\xi)$ denote the objective value of the kth ant/solution and that of the ants/solutions saved in memories, respectively. The variable s_{memory}^{k} represents their associated similarities and δ is a predefined similarity threshold.

(6) *Stopping criteria*.

Whenever the kth ant finishes visiting all the basic nodes, it will move a further step from the last visited basic node according to node transition rule. In such situation, ant k completes a tour and stops.

2.3 API algorithm

Each ant/topology saved in the ant memory (n_m) in the previous stage (AS algorithm) is subsequently implemented for the sizing and shaping optimization using API algorithm separately.

Consider a continuous space S^k for the kth ant/topology and a (n_m+n_n)-dimensional vector solution $s^{k}=\left\{A_1^k,\cdots,A_i^k,\cdots,A_{n_m}^k,P_1^k,\cdots,P_j^k,\cdots,P_{n_n}^k\right\}\in S^k$. A_i^k and P_j^k indicate the cross-sectional area of the ith member and the coordinate of the jth node in the kth ant/topology obtained in the previous stage, respectively. Parameters n_m and n_n denote the number of members and nodes for the kth ant/topology.

First, a nest N^k for the kth ant's colony is randomly created in S^k using O_{rand} operator [5]. Afterward a population of m_k ants leaves the nest to randomly create p_k hunting sites in the neighborhood around N^k utilizes O_{explo} with an amplitude $L_{site}(a_i)$ according to the following equation:

$$\hat{s}^{k}=s^{k}+\boldsymbol{U}[-0.5,+0.5]\cdot L_{site}(a_k)\cdot(s_{\max}^{k}-s_{\min}^{k})$$

$L_{site}(1)$=0.01 and $L_{site}(i)=\left(1/0.01\right)^{i/n}\times 0.01$, i = 2,.., m_k.

where $\boldsymbol{U}$[-0.5, +0.5] denotes a number generated using uniform distribution between [-0.5, 0.5] while $s_{\max}^{k}$, $s_{\min}^{k}$ are the associated maximum and minimum values of s^k.

Subsequently each ant will then locally search for its local optimal solution s_{opt}^{k} nearby its hunting site performing an exploration O_{explo} with an amplitude $L_{local}(a_i)$. A local exploration is successful if it leads to a better evaluation. Then the kth ant memorizes this success and updates the site in its memory from s^k to s_{opt}^{k} . Consequently, sites are erased from the ants' memory after nest N^k move to the best solution found. On the contrary, sites are erased from memory and the location of the nest N^k is unchanged if the local search is failed to find a better solution. Local search procedure continues until the search iteration reaches the defined number t_{local}.

3. Simulation and Discussions

A two-tier, 39-member, 12-node ground structure as Fig. 1 shown is used to evaluate the performance of the proposed algorithm. Material properties are listed below,

Young's modulus = 10^4 ksi,
Density = 0.1 lb/in^3,
Allowable compressive strength = 20 ksi,
Allowable tensile strength = 20 ksi,
Allowable displacement = 2 in,
[$\boldsymbol{A}^{\min}$, $\boldsymbol{A}^{\max}$] = [0, 2.25] in^2.

In addition, coordination of the non-basic nodes are kept as decision variables and assumed to vary within [-120, 120] in. Besides, symmetry about the vertical member at the center of the trusses is employed to reduce number of variables as [1] assumed. Table 1 lists the parameters employed in AS and API algorithms.

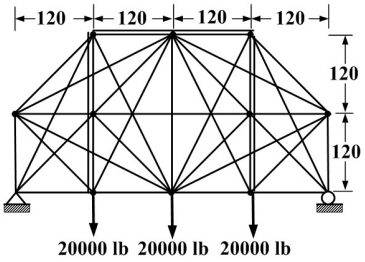


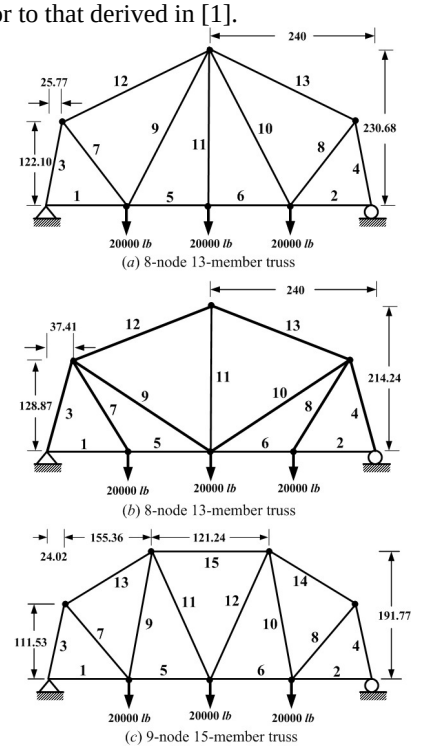
Fig. 1 Two-tier, 39-member, 12-node ground structure

Table 1 Parameters employed in truss simulation

AS algorithm		API algorithm	
Number of iteration n	200	Number of iteration	100
Number of ants m	40	Number of ants m_k	60
Number of ant memory n_m	4	Number of hunting sites p_k	2
Number of elitist ants e	5	Local search times (nest moving)	50
Threshold of similarity δ	0.8	Searching times in hunting sites	30
α	1.0		
ρ	0.2		
τ_0	1×10^{-6}		

Fig. 2(a), 2(b) and 2(c) illustrates the three different truss structures derived in this study (with 191.755 lb, 191.157 lb and 188.732 lb overall weight,

respectively). All these results are lighter than that (192.19 lb) derived in [1] as Fig. 3 depicted. Moreover, Table 2 tabulates the corresponding member areas while Table 3 lists the stress of each member calculated using ANSYS for fair comparison. The results show that the proposed ant algorithms are superior to that derived in [1].



(*a*) 8-node 13-member truss

(*b*) 8-node 13-member truss

(*c*) 9-node 15-member truss

Fig. 2 Optimal truss structures derived in this study

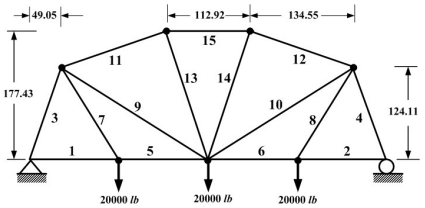


Fig. 3 Optimized truss structures derived in [1]

Table 2 Areas of the optimal truss structures derived in this study by Deb and Gulati [1]

Member number	Areas of members (in^2)		Member number	Areas of members (in^2)	
	Fig. 2(*a*) ant algorithm	Fig. 2(*b*) ant algorithm		Fig. 2(*c*) ant algorithm	Fig. 3 [1]
1, 2	0.322	0.437	**1, 2**	0.327	0.595
3, 4	1.535	1.570	**3, 4**	1.538	1.615
5, 6	1.046	1.077	**5, 6**	1.095	1.166
7, 8	1.218	1.190	**7, 8**	1.221	1.155
9, 10	0.055	0.053	**9, 10**	0.081	0.051
11	1.000	0.945	**11, 12**	0.525	1.293
12, 13	1.189	1.216	**13, 14**	1.259	0.504
			15	1.256	1.358
Weight	**191.755**	**191.157**		**188.732**	**192.19**

Table 3 Stresses of members calculated using ANSYS for the optimized truss structures

Member number	Stress (ksi) calculated using ANSYS		Member number	Stress (ksi) calculated using ANSYS	
	Fig. 2(*a*) ant algorithm	Fig. 2(*b*) ant algorithm		Fig. 2(*c*) ant algorithm	Fig. 3 [1]
1, 2	19.664	19.929	**1, 2**	19.759	19.927
3, 4	-19.975	-19.897	**3, 4**	-19.953	-19.974
5, 6	19.893	19.987	**5, 6**	19.972	19.974
7, 8	19.972	19.962	**7, 8**	19.945	19.960
9, 10	15.206	19.643	**9, 10**	19.920	19.9460
11	20.000	19.981	**11, 12**	19.977	17.288
12, 13	-19.984	-19.994	**13**	-19.977	-19.989
			14, 15	-19.992	-19.921

5. Conclusions

In this paper, a two-stage ant-algorithm based optimization scheme for truss-structure was developed. The topology of the structure is optimized from a given ground structure employing modified AS algorithm while the size and shape of member is optimized utilizing API algorithm. A two-tier, 39-member, 12-node ground structure is used to express the effectiveness of the proposed algorithm compared with the results derived by *Deb* and *Gulati*. The results show that the proposed two-stage ant algorithms are superior to that employed one-stage genetic algorithms.

6. References

[1] K. Deb, and S. Gulati, "Design of truss-structures for minimum weight using genetic algorithms," *Finite Elements in Analysis and Design*, Vol. 37, pp. 447-465, 2001.

[2] O. Hasançebi, and F. Erbatur, "Layout optimization of trusses using simulated annealing," *Advances in Engineering Software*, Vol. 33, pp. 681-696, 2002.

[3] Y. Yang, and CK. Soh, "Automated optimum design of structures using genetic programming," *Computers and Structures*, Vol. 80, pp. 1537-1546, 2002.

[4] M. Dorigo, V. Maniezzo, and A. Colorni, "The ant system: optimization by a colony of cooperating agents," *IEEE Transactions on Systems, Man, and Cybernetics – Part B*, 1996; Vol. 26, pp. 29-41, 1996.

[5] N. Monmarché, G. Venturini, M. Slimane, "On how Pachycondyla apicalis ants suggest a new search algorithm," *Future Generation Computer Systems*, Vol. 16, pp. 937-946, 2000.

[6] CV. Camp, and BJ. Bichon, "Design of space trusses using ant colony optimization," *Journal of Structural Engineering*, Vol. 130, pp. 741-751, 2004.

[7] A. Ghosh, and AK. Mallik, *Theory of Mechanisms and Machines*, affiliated East-West Press, NewDelhi, 1988.